Syntax-based Semi-Supervised Named Entity Tagging

Behrang Mohit
Intelligent Systems Program
University of Pittsburgh
Pittsburgh, PA 15260 USA
behrang@cs.pitt.edu

Rebecca Hwa
Computer Science Department
University of Pittsburgh
Pittsburgh, PA 15260, USA
hwa@cs.pitt.edu

Abstract

We report an empirical study on the role of syntactic features in building a semi-supervised named entity (NE) tagger. Our study addresses two questions: What types of syntactic features are suitable for extracting potential NEs to train a classifier in a semi-supervised setting? How good is the resulting NE classifier on testing instances dissimilar from its training data? Our study shows that constituency and dependency parsing constraints are both suitable features to extract NEs and train the classifier. Moreover, the classifier showed significant accuracy improvement when constituency features are combined with new dependency feature. Furthermore, the degradation in accuracy on unfamiliar test cases is low, suggesting that the trained classifier generalizes well.

1 Introduction

Named entity (NE) tagging is the task of recognizing and classifying phrases into one of many semantic classes such as *persons*, *organizations* and *locations*. Many successful NE tagging systems rely on a supervised learning framework where systems use large annotated training resources (Bikel et. al. 1999). These resources may not always be available for non-English domains. This paper examines the practicality of developing a syntax-based semi-supervised NE tagger. In our study we compared the effects of two types of syntactic rules (constituency and dependency) in extracting and classifying potential named entities. We train a Naive Bayes classification model on a combination of labeled and unlabeled examples with the Expectation Maximization (EM) algorithm. We find that a significant improvement in classification accuracy can be achieved when we combine both dependency and constituency extraction methods. In our experiments, we evaluate the generalization (coverage) of this bootstrapping approach under three testing schemas. Each of these schemas represented a certain level of test data coverage (recall). Although the system performs best on (unseen) test data that is extracted by the syntactic rules (i.e., similar syntactic structures as the training examples), the performance degradation is not high when the system is tested on more general test cases. Our experimental results suggest that a semi-supervised NE tagger can be successfully developed using syntax-rich features.

2 Previous Works and Our Approach

Supervised NE Tagging has been studied extensively over the past decade (Bikel et al. 1999, Baluja et. al. 1999, Tjong Kim Sang and De Meulder 2003). Recently, there were increasing interests in semi-supervised learning approaches. Most relevant to our study, Collins and Singer (1999) showed that a NE Classifier can be developed by bootstrapping from a small amount of labeled examples. To extract potentially useful training examples, they first parsed the sentences and looked for expressions that satisfy two constituency patterns (appositives and prepositional phrases). A small subset of these expressions was then manually labeled with their correct NE tags. The training examples were a combination of the labeled and unlabeled data. In their studies,

Collins and Singer compared several learning models using this style of semi-supervised training. Their results were encouraging, and their studies raised additional questions. First, are there other appropriate syntactic extraction patterns in addition to appositives and prepositional phrases? Second, because the test data were extracted in the same manner as the training data in their experiments, the characteristics of the test cases were biased. In this paper we examine the question of how well a semi-supervised system can classify arbitrary named entities. In our empirical study, in addition to the constituency features proposed by Collins and Singer, we introduce a new set of dependency parse features to recognize and classify NEs. We evaluated the effects of these two sets of syntactic features on the accuracy of the classification both separately and in a combined form (union of the two sets).
Figure 1 represents a general overview of our system's architecture which includes the following two levels: *NE Recognizer* and *NE Classifier.*
Section 3 and 4 describes these two levels in details and section 5 covers the results of the evaluation of our system.

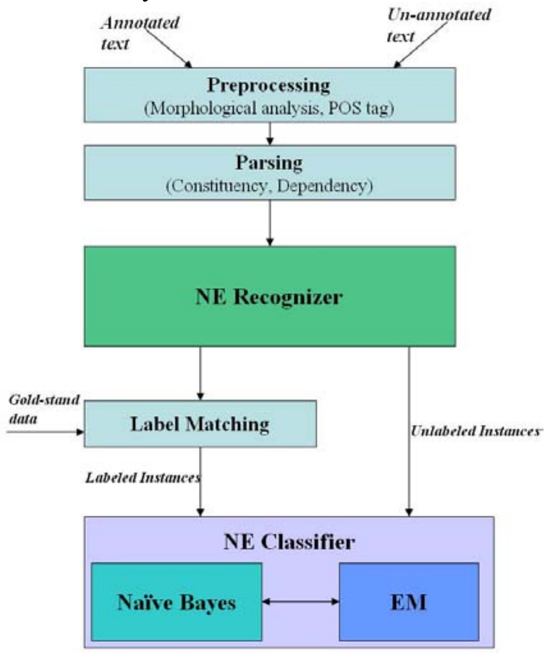

Figure 1: System's architecture

3 Named Entity Recognition

In this level, the system used a group of syntax-based rules to recognize and extract potential named entities from constituency and dependency parse trees. The rules are used to produce our training data; therefore they needed to have a narrow and precise coverage of each type of named entities to minimize the level of training noise.
The processing starts from construction of constituency and dependency parse trees from the input text. Potential NEs are detected and extracted based on these syntactic rules.

3.1 Constituency Parse Features

Replicating the study performed by Collins-Singer (1999), we used two constituency parse rules to extract a set of proper nouns (along with their associated contextual information). These two constituency rules extracted proper nouns within a noun phrase that contained an appositive phrase and a proper noun within a prepositional phrase.

3.2 Dependency Parse Features

We observed that a proper noun acting as the subject or the object of a sentence has a high probability of being a particular type of named entity. Thus, we expanded our syntactic analysis of the data into dependency parse of the text and extracted a set of proper nouns that act as the subjects or objects of the main verb. For each of the subjects and objects, we considered the maximum span noun phrase that included the modifiers of the subjects and objects in the dependency parse tree.

4 Named Entity Classification

In this level, the system assigns one of the 4 class labels *(<PER>, <ORG>, <LOC>, <NONE>)* to a given test NE. The *NONE* class is used for the expressions mistakenly extracted by syntactic features that were not a NE. We will discuss the form of the test NE in more details in section 5. The underlying model we consider is a Naïve Bayes classifier; we train it with the Expectation-Maximization algorithm, an iterative parameter estimation procedure.

4.1 Features

We used the following syntactic and spelling features for the classification:
Full NE Phrase.
Individual word: This binary feature indicates the presence of a certain word in the NE.

Punctuation pattern: The feature helps to distinguish those NEs that hold certain patterns of punctuations like *(...)* for *U.S.A.* or *(&.)* for *A&M*.
All Capitalization: This binary feature is mainly useful for some of the NEs that have all capital letters. such as AP, AFP, CNN, etc.
Constituency Parse Rule: The feature indicates which of the two constituency rule is used for extract the NE.
Dependency Parse Rule: The feature indicates if the NE is the subject or object of the sentence.
Except for the last two features, all features are spelling features which are extracted from the actual NE phrase. The constituency and dependency features are extracted from the NE recognition phase (section 3). Depending on the type of testing and training schema, the NEs might have 0 value for the dependency or constituency features which indicate the absence of the feature in the recognition step.

4.2 Naïve Bayes Classifier

We used a Naïve Bayes classifier where each NE is represented by a set of syntactic and word-level features (with various distributions) as described above. The individual words within the noun phrase are binary features. These, along with other features with multinomial distributions, fit well into Naïve Bayes assumption where each feature is dealt independently (given the class value). In order to balance the effects of the large binary features on the final class probabilities, we used some numerical methods techniques to transform some of the probabilities to the log-space.

4.3 Semi-supervised learning

Similar to the work of Nigam et al. (1999) on document classification, we used Expectation Maximization (EM) algorithm along with our Naïve Bayes classifier to form a semi supervised learning framework. In this framework, the small labeled dataset is used to do the initial assignments of the parameters for the Naïve Bayes classifier. After this initialization step, in each iteration the Naïve Bayes classifier classifies all of the unlabeled examples and updates its parameters based on the class probability of the unlabeled and labeled NE instances. This iterative procedure continues until the parameters reach a stable point. Subsequently the updated Naïve Bayes classifies the test instances for evaluation.

5 Empirical Study

Our study consists of a 9-way comparison that includes the usage of three types of training features and three types of testing schema.

5.1 Data

We used the data from the Automatic Content Extraction (ACE)'s entity detection track as our labeled (*gold standard*) data.[1]
For every NE that the syntactic rules extract from the input sentence, we had to find a matching NE from the gold standard data and label the extracted NE with the correct NE class label. If the extracted NE did not match any of the gold standard NEs (for the sentence), we labeled it with the <*NONE*> class label.
We also used the WSJ portion of the Penn Tree Bank as our unlabeled dataset and ran constituency and dependency analyses[2] to extract a set of unlabeled named entities for the semi-supervised classification.

5.2 Evaluation

In order to evaluate the effects of each group of syntactic features, we experimented with three different training strategies (using constituency rules, dependency rules or combinations of both). We conducted the comparison study with three types of test data that represent three levels of coverage (recall) for the system:
1. Gold Standard NEs: This test set contains instances taken directly from the ACE data, and are therefore independent of the syntactic rules.
2. Any single or series of proper nouns in the text: This is a heuristic for locating potential NEs so as to have the broadest coverage.
3. NEs extracted from text by the syntactic rules. This evaluation approach is similar to that of Collins and Singer. The main difference is that we have to match the extracted expressions to a pre-

[1] We only used the NE portion of the data and removed the information for other tracking and extraction tasks.
[2] We used the Collins parser (1997) to generate the constituency parse and a dependency converter (Hwa and Lopez, 2004) to obtain the dependency parse of English sentences.

labeled gold standard from ACE rather than performing manual annotations ourselves.

All tests have been performed under a 5-fold cross validation training-testing setup. Table 1 presents the accuracy of the NE classification and the size of labeled data in the different training-testing configurations. The second line of each cell shows the size of *labeled* training data and the third line shows the size of testing data. Each column presents the result for one type of the syntactic features that were used to extract NEs. Each row of the table presents one of the three testing schema. We tested the statistical significance of each of the cross-row accuracy improvements against an alpha value of 0.1 and observed significant improvement in all of the testing schemas.

Testing Data	**Training Features**		
	Const.	**Dep.**	**Union**
Gold Standard NEs (ACE Data)	76.7% 668 579	78.5% 884 579	82.4% 1427 579
All Proper Nouns	70.2% 668 872	71.4% 884 872	76.1% 1427 872
NEs Extracted by Training Rules	78.2% 668 169	80.3% 884 217	85.1% 1427 354

Table 1: Classification Accuracy, labeled training & testing data size

Our results suggest that dependency parsing features are reasonable extraction patterns, as their accuracy rates are competitive against the model based solely on constituency rules. Moreover, they make a good complement to the constituency rules proposed by Collins and Singer, since the accuracy rates of the union is higher than either model alone. As expected, all methods perform the best when the test data are extracted in the same manner as the training examples. However, if the systems were given a well-formed named entity, the performance degradation is reasonably small, about 2% absolute difference for all training methods. The performance is somewhat lower when classifying very general test cases of all proper nouns.

6 Conclusion and Future Work

In this paper, we experimented with different syntactic extraction patterns and different NE recognition constraints. We find that semi-supervised methods are compatible with both constituency and dependency extraction rules. We also find that the resulting classifier is reasonably robust on test cases that are different from its training examples.

An area that might benefit from a semi-supervised NE tagger is machine translation. The semi-supervised approach is suitable for non-English languages that do not have very much annotated NE data. We are currently applying our system to Arabic. The robustness of the syntactic-based approach has allowed us to port the system to the new language with minor changes in our syntactic rules and classification features.

Acknowledgement

We would like to thank the NLP group at Pitt and the anonymous reviewers for their valuable comments and suggestions.

References

Shumeet Baluja, Vibhu Mittal and Rahul Sukthankar, 1999. Applying machine learning for high performance named-entity extraction. In *Proceedings of Pacific Association for Computational Linguistics.*

Daniel Bikel, Robert Schwartz & Ralph Weischedel, 1999. An algorithm that learns what's in a name. Machine Learning 34.

Michael Collins, 1997. Three generative lexicalized models for statistical parsing. *In Proceedings of the 35th Annual Meeting of the ACL.*

Michael Collins, and Yoram Singer, 1999. Unsupervised Classification of Named Entities. *In Proceedings of SIGDAT.*

A. P. Dempster, N. M. Laird and D. B. Rubin, 1977. Maximum Likelihood from incomplete data via the EM algorithm. *Journal of Royal Statistical Society, Series B, 39(1), 1-38.*

Rebecca Hwa and Adam Lopez, 2004. On the Conversion of Constituent Parsers to Dependency Parsers. *Technical Report TR-04-118, Department of Computer Science, University of Pittsburgh.*

Kamal Nigam, Andrew McCallum, Sebastian Thrun and Tom Mitchell, 2000. Text Classification from Labeled and Unlabeled Documents using EM. *Machine Learning 39(2/3).*

Erik F. Tjong Kim Sang and Fien De Meulder, 2003. Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. *In Proceedings of CoNLL-2003.*